Specialized Computing and Robotics Language (SCARL) Grammar *(first draft)*

program → *statement_list*

statement_list → *statement* | *statement statement_list*

statement → *device_declarator_statement* | *primitive_definition_statement* | *function_definition_statement*

block_statment → **'{'** *statement_list_block_level* **'}'**

statement_list_block_level → *statement_block_level* | *statement_block_level statement_list_block_level*

statement_block_level → *primitive_definition_statement* | *block_statement* | *variable_set_statement* |
function_invocation_statement | *if_block_statement* | *while_block_statement*

device_declarator_statement → *device_type* **'IDENTIFIER' ';'**

primitive_declarator → *primitive_type* **'IDENTIFIER'**

primitive_definition_statement → *primitive_declarator* **'='** *expression* **';'**

function_definiton_statement → *primitive_declarator* **'('** *formal_parameter_list* **')'**
block_statement

variable_set_statement → **'IDENTIFIER' '='** *expression* **';'**

function_invocation → **'IDENTIFIER' '('** *parameter_list* **')'**

function_invocation_statement → *function_invocation* **';'**

if_block_statement → **'if' '('** *expression* **')'** *block_statement* |
'if' '(' *expression* **')'** *block_statement* **'else'** *block_statement*

while_block_statement → **'while' '('** *expression* **')'** *block_statement*

formal_parameter_list → *primitive_declarator* |
primitive_declarator **','** *formal_parameter_list* | *ε*

parameter_list → *expression* | *expression* **','** *parameter_list* | *ε*

expression → *logical_expression*

logical_expression → *logical_and_expression* |
logical_expression **'||'** *logical_and_expression*

logical_and_expression → *equality_expression* |
logical_and_expression **'&&'** *equality_expression*

equality_expression → *relational_expression*|
equality_expression **'=='** *relational_expression* |
equality_expression **'!='** *relational_expression*

relational_expression → *bool_expression* |
relational_expression **‘>’** *bool_expression* |
relational_expression **‘<’** *bool_expression* |
relational_expression **‘>=’** *bool_expression* |
relational_expression **‘<=’** *bool_expression*

bool_expression → *arithmetic_expression* |
‘!’ *arithmetic_expression*

arithmetic_expression → *arithmetic_factor* |
arithmetic_expression **‘+’** *arithmetic_factor* |
arithmetic_expression **‘-‘** *arithmetic_factor*

arithmetic_factor → *arithmetic_unary* |
arithmetic_factor **’*’** *arithmetic_unary* |
arithmetic_factor **‘/’** *arithmetic_unary*

arithmetic_unary → *unit* | **‘-‘** *arithmetic_unary* | **‘(‘** *arithmetic_expression* **‘)’**

unit → **‘IDENTIFIER’** | *integer_value* | *bool_value* | *function_invocation*

integer_value → **‘DECIMAL’** | **‘OCTAL’** | **‘HEX’** | **‘BINARY’**

bool_value → **‘true’** | **‘false’**

primitive_type → **‘bool’** | **‘int’** | **‘char’** | **‘pointer’** | **‘void’**

device_type → **‘LightActuator’** | **‘ServoActuator’** | **’SoundSensor’** |
‘LightSensor’ | **‘DistanceSensor’** | **‘TemperatureSensor’**